Dynamic Bandwidth Adjusting in BlockDAG Networks

Sergii Grybniak[1], s.s.grybniak@op.edu.ua Yevhen Leonchyk[2], leonchyk@onu.edu.ua
Igor Mazurok[2], mazurok@onu.edu.ua Oleksandr Nashyvan[1], o.nashyvan@op.edu.ua
Alisa Vorokhta[2], alisa.vorokhta@stud.onu.edu.ua

1. Institute of Computer Systems, Odesa Polytechnic National University, Ukraine
2. Faculty of Mathematics, Physics and Information Technologies, Odesa I.I. Mechnikov National University, Ukraine

Abstract. This work presents a mechanism for dynamically adapting a decentralized network to fluctuations in transaction flow, based on a directed acyclic graph (DAG) ledger structure. By incorporating this approach, a system can effectively respond to changing network workloads, ensuring a self-sufficient and adaptive environment for processing a transaction throughout its entire lifespan. Although this dynamic adoption mechanism is designed explicitly for the current version of the Waterfall platform, the presented approach possesses the potential to be applied across a wide range of networks built on the blockDAG principle, owing to a set of tuning parameters.

Keywords – distributed ledger technology, directed acyclic graph, blockDAG, system dynamic adaptation, scalability.

Introduction

As the adoption of Distributed Ledger Technology (DLT) expands [1], the issue of scalability poses a significant challenge, where systems need to handle a growing number of transactions and users [2, 3]. This is especially pronounced in the enterprise sector. Thus, only by providing scalability can DLT unlock its full potential as a transformative technology, capable of supporting a global decentralized infrastructure for secure and efficient digital transactions.

At present, many networks proposing solutions to overcome the scalability limitations of distributed systems suggest a tradeoff with decentralization. However, the emerging Directed Acyclic Graph (DAG) ledger design can be considered the next generation of blockchain, able to solve the scalability problem while preserving decentralization [4]. Unlike blockchain technology with linear sequenced blocks, in blockDAG-based systems, the timeline is composed of distinct slots, each of which can have multiple blocks. Those blocks form referential structures (fig. 1) since they can (and sometimes must) reference several previous blocks [5, 6]. In other words, when a new block is created, the block signer records into it references to some blocks from previous slots, called tips. Tip-blocks are blocks that are not referenced by any other blocks known to the block signer at the moment.

Figure 1: An example of a starting fragment of a blockDAG created using simulation.

On the Waterfall platform [7] and other networks of a similar type, as a result of operations, the created blocks of transactions line up in a DAG, part of which is topologically sorted and finalized in a so-called Stream (fig. 2). The remaining part, consisting of not-yet-ordered blocks, forms a Spray. The width of the Spray area is the maximum possible number of blocks per slot characterizing network bandwidth. Since finalized blocks of the same slot (a vertical line in fig. 2) are created in parallel, it directly affects the current level of scalability.

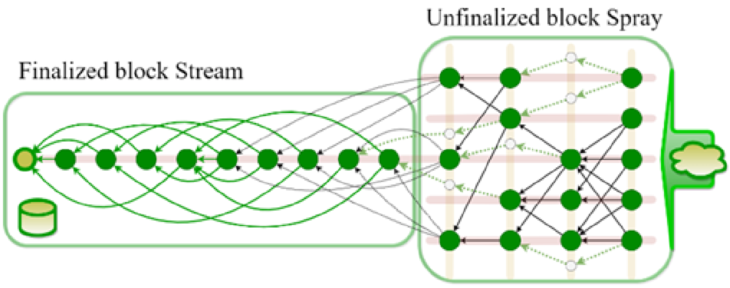

Figure 2: The Stream and Spray areas

Obviously, the network workload varies over time, depending on a lot of off-chain factors. Therefore, a system should increase the width of the Spray area to process more transactions, and vice versa, decreasing the width so as not to create empty blocks. The primary goal of this work is to present a mechanism for dynamic adaptation of the Spray width, enabling the system to attain self-sufficiency throughout its entire lifespan.

Automatically Adjusting Algorithm

Referential Structure Status

Generally, a mandatory task for a DAG-based system is to maintain a valid referential structure. Referential integrity ensures consistent and reliable linkage of blocks by accurately referencing their respective previous blocks, maintaining the network's validity and security. It could be considered good in a blockDAG network when all blocks are properly and unambiguously linked to enough respective previous blocks, ensuring the absence of conflicts, double-spending, and other security vulnerabilities. The algorithm for recalculating the referential structure is executed upon receiving each new block and has a complexity of $O(n)$, where n is the number of references within it.

In the ideal case (100% of integrity), where all blocks propagate across the network in time, each block in a slot refers to all blocks in the previous slot. In practice, the simultaneous creation and distribution of a large number of blocks leads to an increase in network traffic, causing block propagation delays. Delays in turn lead to a deterioration in the referential structure, since the number of links may decrease and/or be referred to tip-blocks that are located in earlier slots than the previous one. Therefore, when increasing the number of blocks in one slot, it is necessary to keep the link structure at an acceptable level.

For this purpose, according to the results of each epoch (a specific period of time, e.g. 32 slots in Waterfall), the values are determined:

$$S_0 = p_1 \sum_{i=2}^{E} b_{i-1} b_i \qquad \text{and} \qquad S = \sum_{k=1}^{B} \sum_{j=1}^{t_k} p_{l_{kj}},$$

where E is the slot number per epoch, b_i is the block number in i-th slot (this varies since some blocks could be missed for different reasons), B is the block number in the epoch, t_k is the number of tips in k-th block, l_{kj} is the connection depth of k-th block with its j-th tip, and the vector $p = \left(p_1, p_2, \dots, p_n\right)$ is weights that correspond to depth indeces. Here S_0 reflects the ideal case mentioned above, and S presents the referential structure status during the review epoch.

Hence, the ratio $R = S/S_0$ in comparison with a pre-selected reference point $R_0 \in (0; 1)$ can be used to estimate the status of the epoch referential structure. The appropriate value of R_0 should be defined for each DAG-based network separately, depending on its own characteristics. Further, the relation R will be applied to set an upper boundary on the Spray width for the next epoch (W). If it is not possible to obtain R and R_0 or to simplify the management model, the maximum of W could be bounded with a fixed value.

Spray Width

Let L be the total workload for an epoch. In the Waterfall network, this value is measured in consumed units of gas. For example, one ordinary transaction takes approximately 21,000 gas units, but transactions calling for a smart contract can require far more units of gas. The Spray width for the next epoch can be obtained on the basis of the workload for the just-completed epoch by the naive forecasting method [8]:

$$W = max\left(W_{min}, \left[\frac{L}{C \cdot E \cdot \alpha}\right]\right),$$

where W_{min} is the constant that bounds the width from below, C is the block capacity (210 million gas in Waterfall), with the number E of slots per epoch, $\alpha \in (0; 1)$ is the desired level of block occupancy, and $[\cdot]$ means rounding. A value of $\alpha < 1$ in the formula's denominator provides additional block space for the next epoch, and also allows us to take into account workload fluctuations between slots of the same epoch. However, with $R > R_0$, an increase in Spray width is not performed in order to avoid worsening the referential structure. Even if the above formula indicates this, the width can be decreased anyway.

To make this happen, all nodes must have the same opinion about the previous epoch workload. Hence, only finalized epochs recorded into the ledger can be considered, since such data are always consistent and accessible to everyone involved. In Waterfall, a value of L is calculated based on the last finalized epoch. Additionally, more sophisticated forecasting methods, e.g. technical analysis, can be used to estimate the next epoch workload, and consequently the value of its Spray width.

Simulation

Network Workload

When a new product or service is introduced, the rate of new customer acquisition and, as a result, its utilization, rises at first. However, as the market saturates, that rate slows down as the product or service reaches its full potential. Thus, to simulate the network load, it is advisable to use a logistic equation that is a mathematical function characterized by increasing growth in the beginning period, but decreasing growth at a later stage

$$L(t) = \frac{L_{max}}{1+exp(-q(t-t_0))}, \qquad t \geq 0,$$

where L_{max} is the supremum (the value of network saturation), q is the steepness of the curve, and t_0 is the inflection point (the curve's midpoint).

The logistic function is a powerful tool for mathematical modeling across many fields and can provide valuable insights into the dynamics of complex systems over time [9]. In our case, we determine the network workload $L(t)$ as the amount of computation (so-called gas) needed to process all users' transactions per epoch.

Results

The study utilized statistical analysis techniques, as well as simulation modeling experiments in Python. Network workloads over wide ranges of L_{max}, q, and t_0 parameters were tested. In doing so, we investigated both a workload increase ($q > 0$) and its decrease ($q < 0$). In all cases, the method demonstrated a good response to changing conditions. For example, figure 3 depicts a mixed case with the desired level of block occupancy as $\alpha = 0.5$ and workload parameters as $L_{max} = 134.4$ billion gas (which corresponds to 640 completely filled blocks per epoch), $t_0 = 123\,272$ epoch (which corresponds to half a year), $q = 5 \cdot 10^{-6}$ for the increasing part, and $t_0 = 246\,544$ epoch (which corresponds to one year), $q = -5 \cdot 10^{-6}$ for the decreasing part. As can be seen, the Spray width equaled $W_{min} = 4$ at first and was further adopted to the varying workload, while the block occupancy reached the value of $\alpha = 0.5$, fluctuating close to it during the testing period.

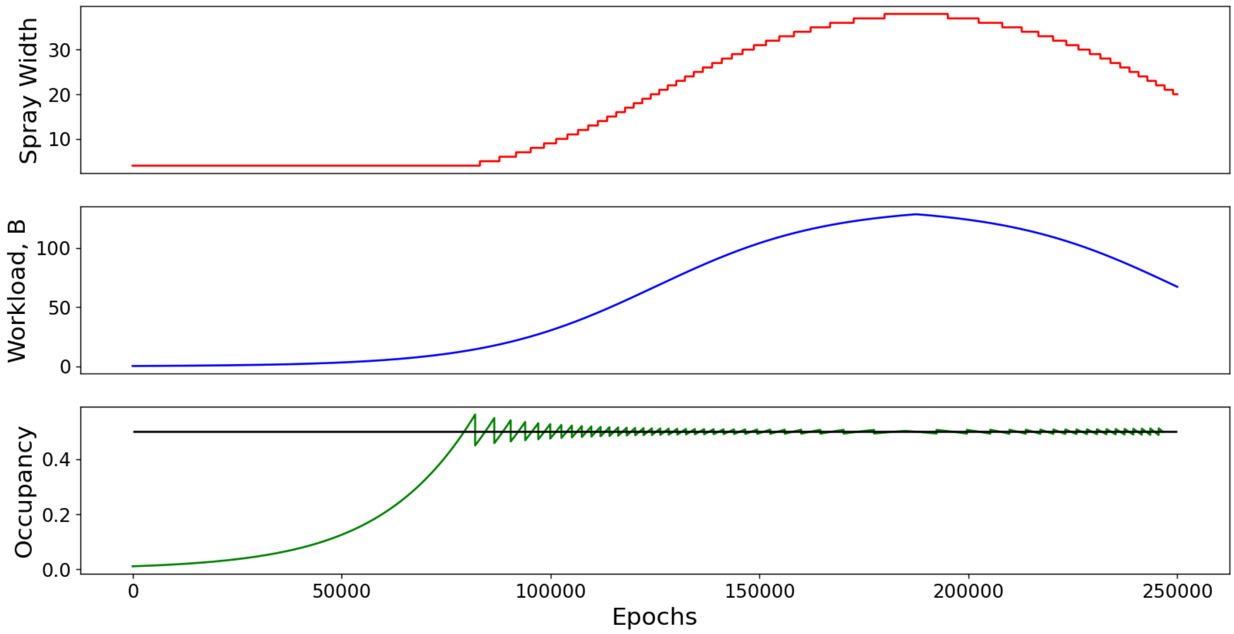

Figure 3: The Spray width, the network workload and the block occupancy by epochs.

Conclusion

Addressing the scalability challenge in DLT is crucial for its widespread adoption and usability. Nowadays, researchers, developers, and industry stakeholders are actively exploring various solutions to enhance scalability, one of which is a blockDAG ledger structure. The blockDAG structure opens up new unique opportunities that differ from those offered by traditional blockchain technology.

In particular, automatically maintaining a high transaction throughput allows us to keep transaction fees low as the network expands in various scenarios, even during peak times. At the same time, when the transaction flow is low, the number of blocks per slot decreases, which prevents low block occupancy. It is worth noting that to make this happen, the dynamic adjustment must also be supported by a consensus protocol and a tokenomics model, e.g. as is being carried out on the Waterfall platform [10-12].

The presented flexible approach allows for easy modification through a carefully tuned set of parameters. This adaptability is crucial for emerging public platforms having different characteristics, as they must be able to evolve effectively under varying workloads. However, obtaining the optimal parameter values poses a sophisticated challenge that requires extensive research, including mandatory practical tests. Only through such comprehensive investigations can we attain a balanced system capable of sustained performance and stability over time.

References

1. A. Perdana, A. Robb, V. Balachandran, & F. Rohde, “Distributed ledger technology: Its evolutionary path and the road ahead,” Information & Management, 58(3), 2021, 103316.
2. J. Xie, F. R. Yu, T. Huang, R. Xie, J. Liu, & Y. Liu, “A survey on the scalability of blockchain systems,” IEEE Network, 2019, pp. 166-173.
3. Q. Zhou, H. Huang, Z. Zheng, & J. Bian, “Solutions to scalability of blockchain: A survey,” IEEE Access, 2020, pp. 16440-16455.
4. F. M. Benčić & I. Podnar Žarko, “Distributed Ledger Technology: Blockchain Compared to Directed Acyclic Graph,” IEEE 38th International Conference on Distributed Computing Systems, 2018, pp. 1569-1570.
5. H. Pervez, M. Muneeb, M. U. Irfan, & I. U. Haq, "A Comparative Analysis of DAG-Based Blockchain Architectures," 12th International Conference on Open Source Systems and Technologies (ICOSST), 2018, pp. 27-34.
6. Q. Wang, J. Yu, S. Chen, & Y. Xiang, “SoK: Diving into DAG-based blockchain systems,” preprint arXiv:2012.06128, 2020.
7. S. Grybniak, D. Dmytryshyn, Y. Leonchyk, I. Mazurok, O. Nashyvan & R. Shanin, “Waterfall: A Scalable Distributed Ledger Technology,” IEEE 1st Global Emerging Technology Blockchain Forum: Blockchain & Beyond (iGETblockchain), Irvine, CA, USA, 2022, pp. 1-6.
8. R. J. Hyndman & G. Athanasopoulos, “Forecasting: principles and practice (3rd ed.),” OTexts, 2021, 442 p.
9. M. A. Razzaque & C. C. Sheng, “Outsourcing of logistics functions: a literature survey,” International Journal of Physical Distribution & Logistics Management, 28(2), 1998, pp. 89-107.
10. S. Grybniak, Y. Leonchyk, I. Mazurok, O. Nashyvan, & R. Shanin, “Waterfall: Gozalandia. Distributed protocol with fast finality and proven safety and liveness,” IET Blockchain, 2023, pp. 1-12.
11. S. Grybniak, Y. Leonchyk, R. Masalskyi, I. Mazurok and O. Nashyvan, "Waterfall: Salto Collazo. Tokenomics," IEEE International Conference on Blockchain, Smart Healthcare and Emerging Technologies (SmartBlock4Health), Bucharest, Romania, 2022, pp. 1-6.
12. S. Grybniak, Y. Leonchyk, I. Mazurok, O. Nashyvan, & A. Vorokhta, “Waterfall: Salto Collazo. High-Level Design of Tokenomics,” Advances in Science, Technology and Engineering Systems Journal, 8(3), 2023, pp. 231-243.